Naval Postgraduate School
Center for INFOSEC Studies and Research

Data Integrity Limitations in Highly Secure Systems

International Systems Security Engineering Conference
Orlando, Florida
March 1, 2001

Cynthia Irvine and Timothy Levin
Center for INFOSEC Studies and Research
Computer Science Department
Naval Postgraduate School, Monterey, California, USA
irvine(levin)@cs.nps.navy.mil

Sponsorship

- This work was sponsored in part by:
  - the Community Intelligence Office and
  - the DARPA/ITO Quorum program

Outline

- Background
  - Integrity
  - High Assurance Systems
- Hybrid Security Architectures
- Integrity and HSAs
- Conclusion

Integrity

- *Dual* of Confidentiality
  - Labels indicate potential loss from
    - unauthorized modification
    - vs. unauthorized disclosure
- Translation from Confidentiality
  - inverted/convoluted
  - difficult concepts [Gasser]
- Analysis can be overlooked

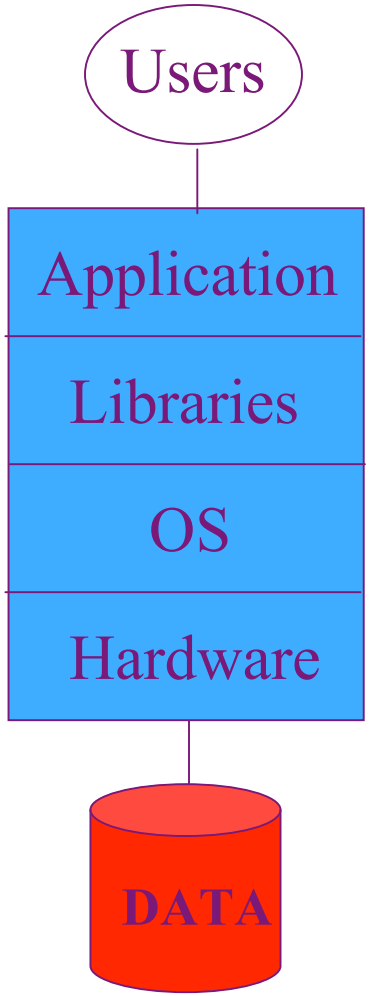
Integrity (2)

- **Inherent** Integrity and Confidentiality
  - explicit labels
  - or implicitly understood
- Unclear distinctions/assumptions
  - High confidentiality implies high integrity?
- Integrity of *Code*
  - fidelity to original - e.g., distributed version
  - fidelity to described intent
    - correct functionality
  - no additional functionality
    - trap doors, Trojan horses

High Assurance Systems

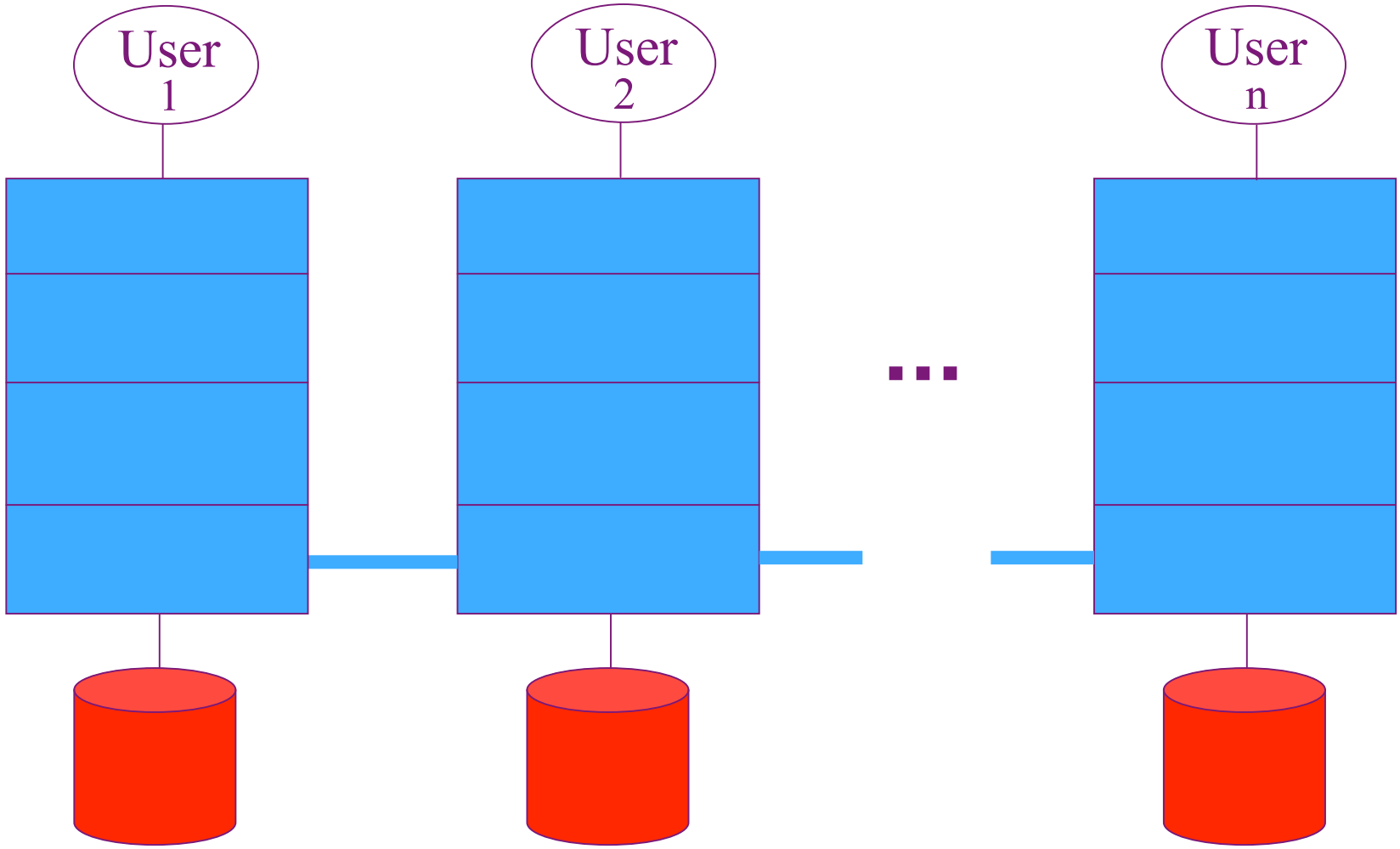
- Enforce confidentiality and integrity
  - to defined degree of assurance
- Various architectural approaches...

In the Beginning...



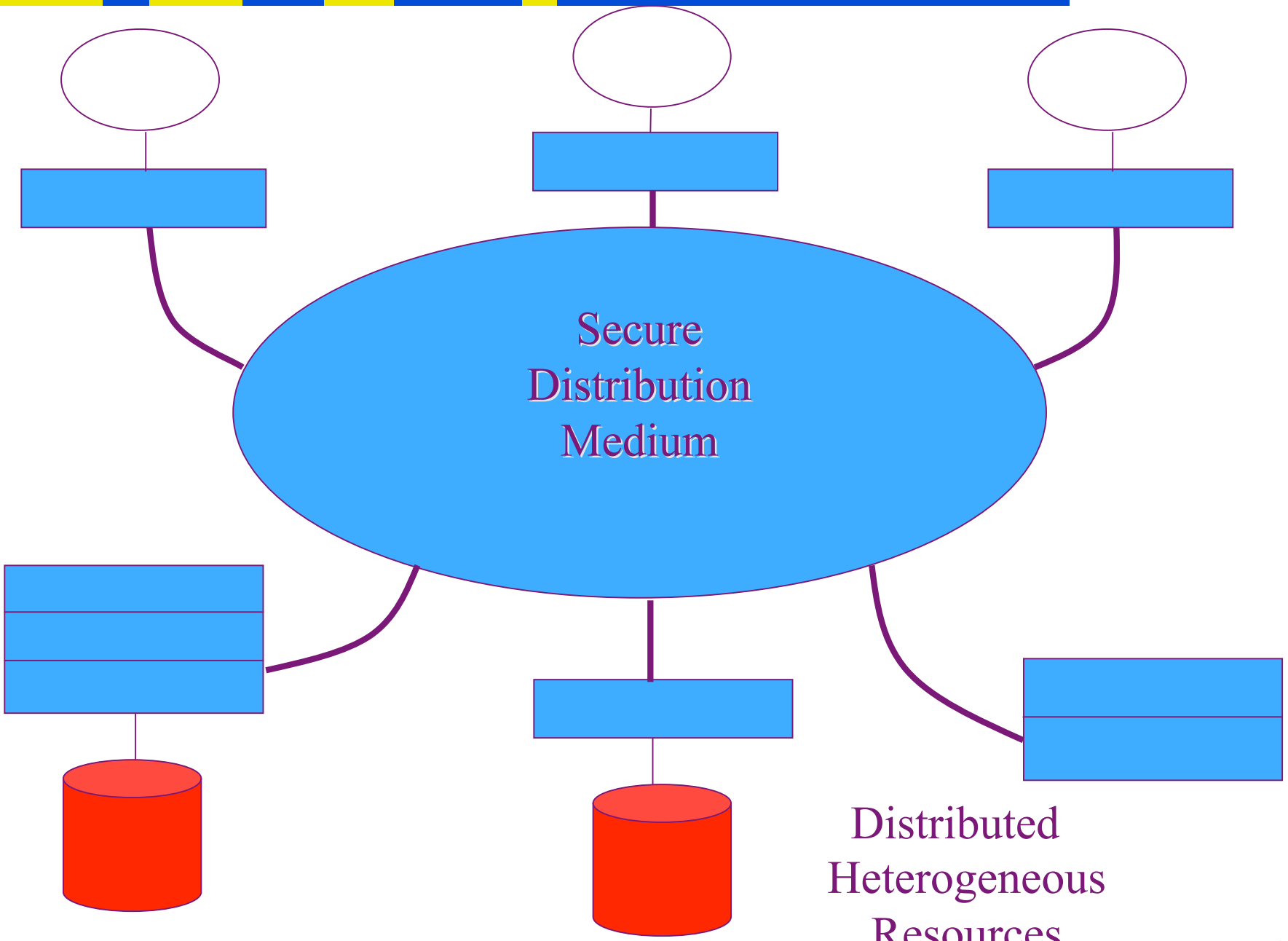
"Secure"

Distributed



Need to Generalize Remote Access

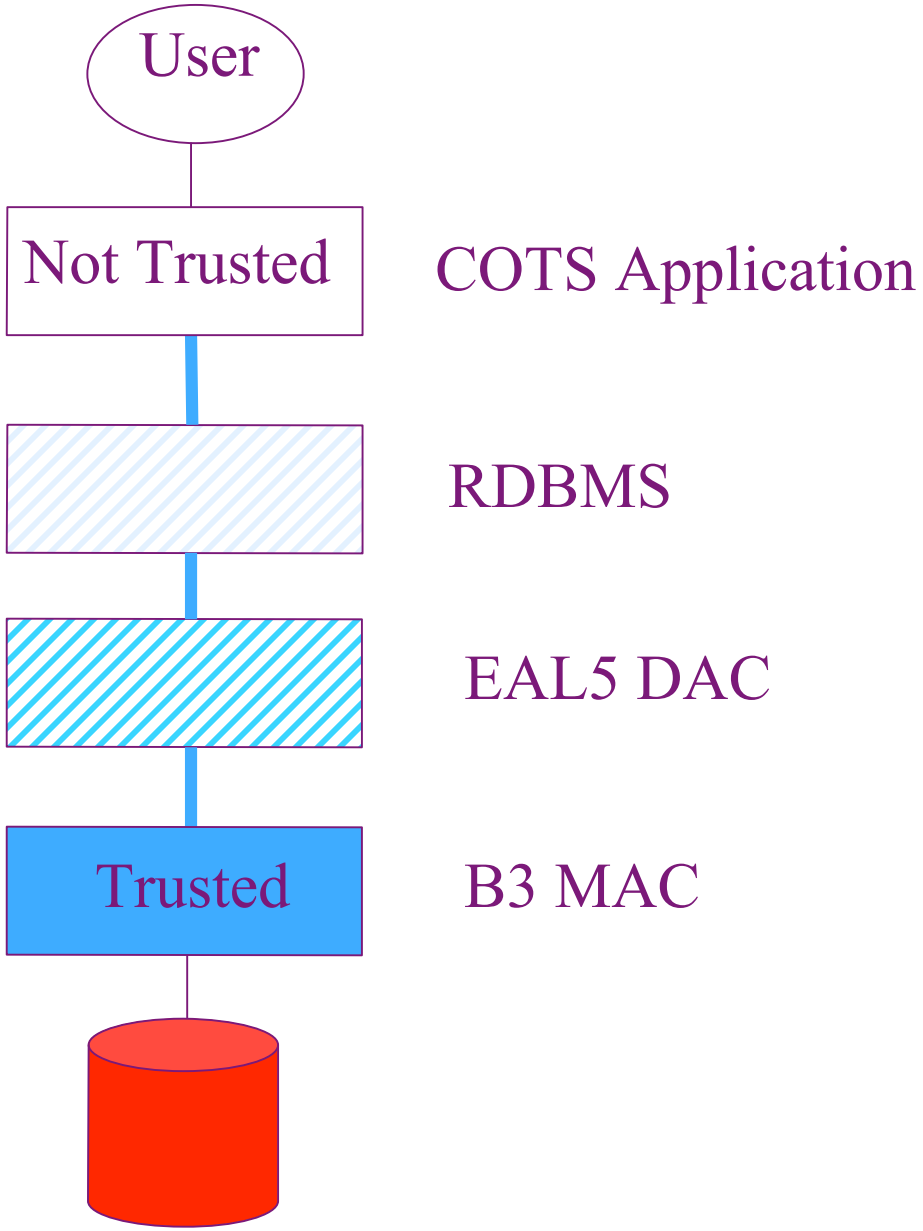
- remote programs
- remote data
- remote processors, devices


Secure
Distribution
Medium
Distributed
Heterogeneous
Resources

High Assurance Systems

- Expensive
- Incompatible, “stovepipes”
- Responses
  - COTS in Government RFPs
  - Balanced Assurance (at vendor initiative?)
  - TCB subsets

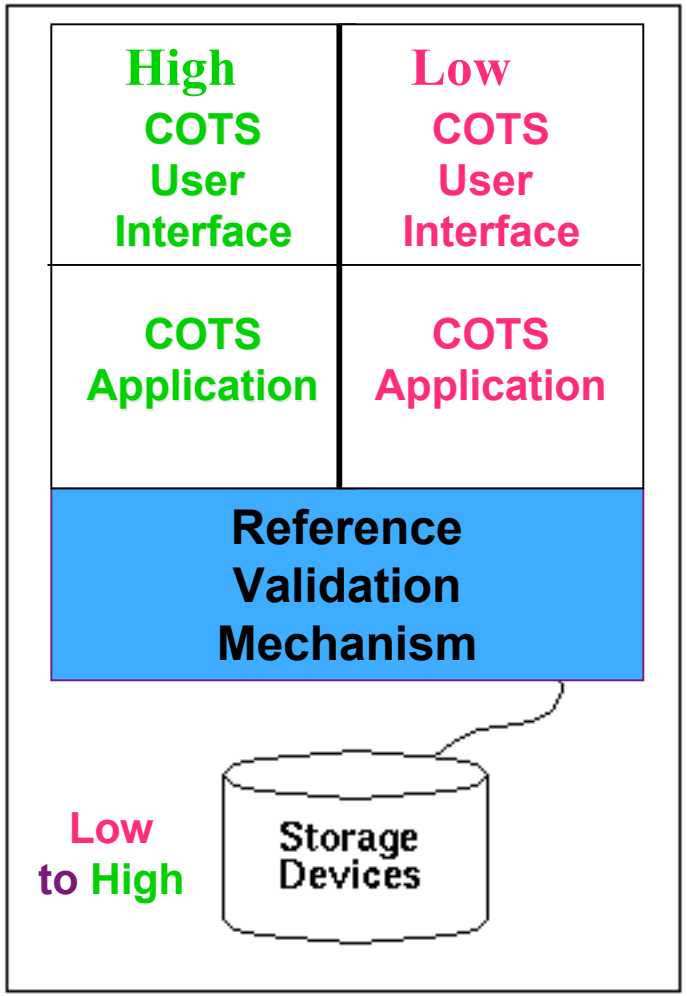
Balanced Assurance



Hybrid Security Architecture

- Balanced Assurance + COTS + MLS
- Configuration Components
  - Untrusted COTS terminals/workstations
  - Untrusted COTS applications
  - Storage devices
    - Multilevel data
  - Multilevel TCB mechanisms (RVM)
  - TCB extensions
  - Network Connections
    - single and multilevel

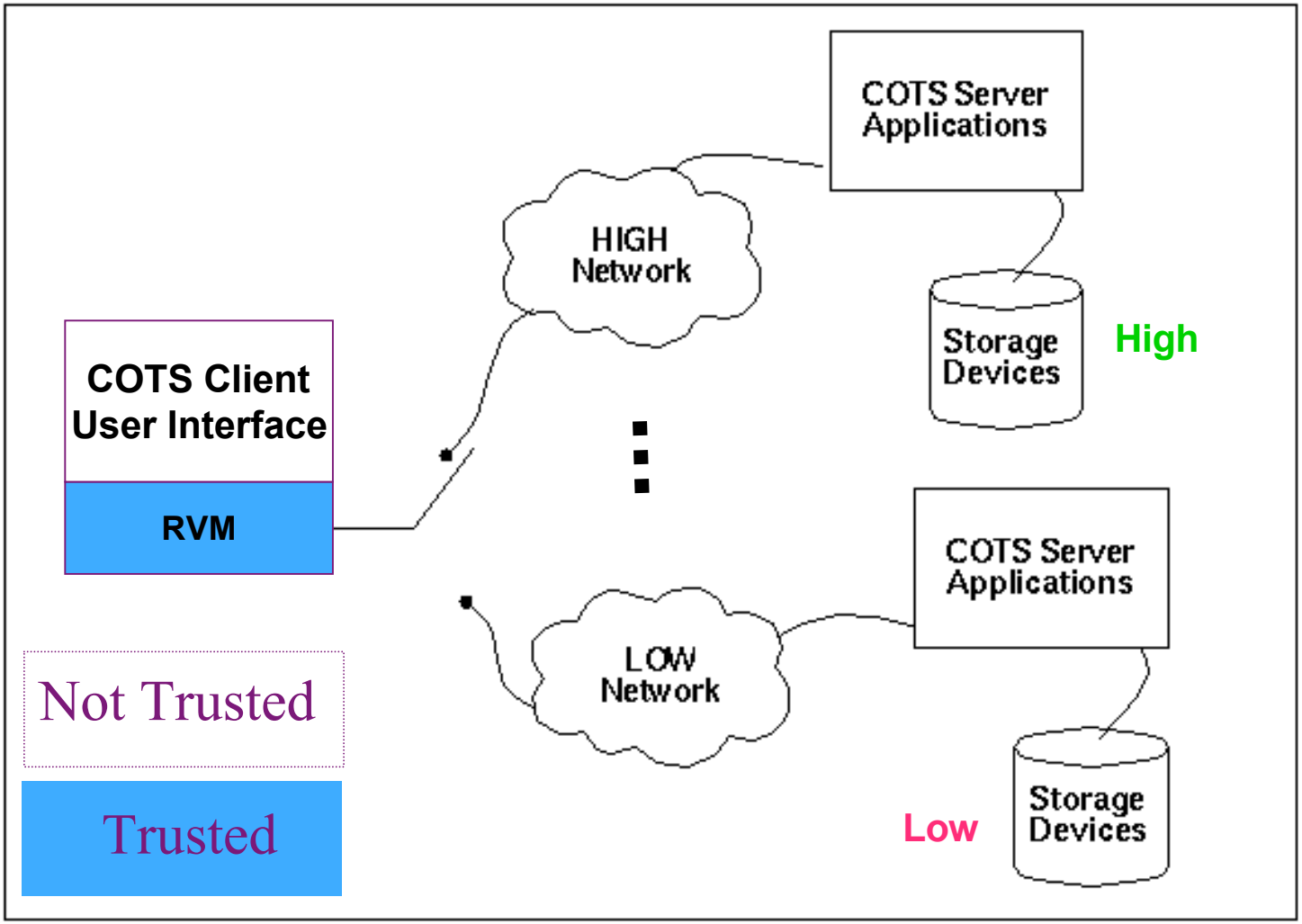
Monolithic



Not Trusted

Trusted

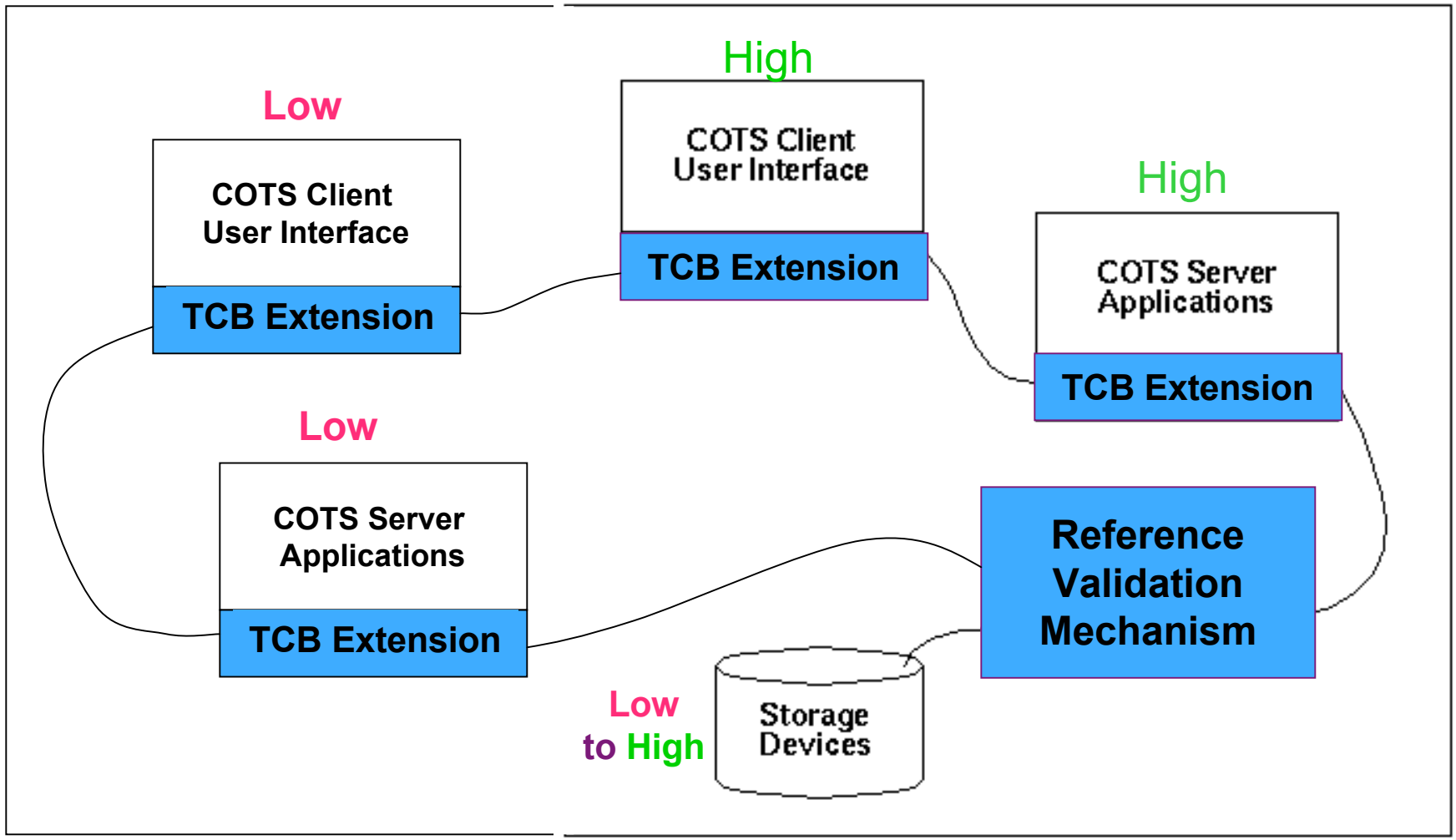
Switch-Based



Distributed

Trusted

Not Trusted



MLS Access Enforcement

- Reference Validation Mechanism
  - mediates access to objects
  - controls object creation, storage, access, I/O
  - prevents data *leakage* across MAC partitions
    - confidentiality write-down or read-up
    - integrity read-down or write-up (if enforced)
- Other modules constrained by RVM
  - leakage

System Confidentiality *Capacity*

- System trusted for confidentiality
  - to confidentiality capacity of reference validation module
    - capacity is relative to RVM assurance
    - Yellow Book
      - Maps assurance levels to confidentiality ranges
    - Policy enforced regardless of untrusted components

Data Integrity

- Integrity Semantics
  - dual of confidentiality
    - "prevents data contamination from untrusted software"
  - was the modification correct?
    - Within the partition
    - Code trusted to handle data correctly
      - to its level of assurance
    - No Yellow Book for integrity
    - Look to code integrity label

Code Module Integrity Label

- What the system designer needs it to be
  - Coherent network architecture
    - least privilege
  - Limit: pedigree of code

Trust in Commercial Applications

- Evaluation below B2/EAL5
  - little config. mgt. or code review required
    - no examination for Trojan horses/trap doors
    - no code correspondence
  - no trusted distribution
  - potential for unknown functionality
    - e.g., “Easter eggs” common in commercial software
    - testing doesn’t address unknowns
- Integrity is “low assurance” or “untrusted”
  - integrity label

System Integrity Capacity

- Some Code Modules can modify user data
  - set of Data Modifying Modules = *DMM*
- System integrity capacity:
  - integrity of **least-trusted** *Module*
    - I_capacity = $GLB_{(m \in DMM)}$ (integrity(m))
  - (system confidentiality capacity is:
    - capacity of reference validation module)

Integrity Capacity: two cases

- Integrity not supported
  - system can take in data higher in integrity than system
  - data output is lowered to integrity GLB of *DMM*
    - de facto label
- Integrity Supported by RVM
  - system regulates its own integrity capacity
    - ∀ m ∈ DMM, o: object (
      - » write (m,o) ⇒ dominates(integrity(m), integrity(o))
    - cannot take in data higher than integrity GLB of DMM
    - assumes modules/subjects are labeled correctly
    - problem not addressed by ring mechanisms

HSA and Integrity

- HSA applications and user interfaces
  - COTS
  - below B2/EAL5 (integrity *untrusted* or *low assurance*)
  - generally designed to modify data
- HSA systems have untrusted or low assurance integrity capacity
- Hybrid Security Architecture systems not suitable for environments with *trusted/critical* data integrity requirements

Summary

- Correct integrity labeling of code is critical
- *Code-module integrity* limits *system integrity capacity*
  - Not new information
  - Not always remembered
  - Not always communicated to sponsors and customers
- HSA systems not suitable in environments w/ *critical or trusted* data

Questions?

- Irvine, levin @cs.nps.navy.mil
- http://cisr.nps.navy.mil

Integrity Labels

- Subject consists of a set of code modules
- ∀ m: module, s: subject o: object(
  write(s,o) & m ∈ s
  ⇒ integrity (m) ≥ integrity(s) ≥ integrity(o))

≥ - enforced by design/configuration

≥ - enforced by RVM

Need both for coherent integrity enforcement

- (Dominates “≥”)

Vertically Distributed